```
MMM         MMM     000000000     NNN         NNN  TTTTTTTTTTTTTTT     000000000     RRRRRRRRRRRR
MMM         MMM     000000000     NNN         NNN  TTTTTTTTTTTTTTT     000000000     RRRRRRRRRRRR
MMM         MMM     000000000     NNN         NNN  TTTTTTTTTTTTTTT     000000000     RRRRRRRRRRRR
MMMMMM   MMMMMM  000         000  NNN         NNN        TTT        000         000  RRR         RRR
MMMMMM   MMMMMM  000         000  NNN         NNN        TTT        000         000  RRR         RRR
MMMMMM   MMMMMM  000         000  NNN         NNN        TTT        000         000  RRR         RRR
MMM   MMM   MMM  000         000  NNNNNN      NNN        TTT        000         000  RRR         RRR
MMM   MMM   MMM  000         000  NNNNNN      NNN        TTT        000         000  RRR         RRR
MMM   MMM   MMM  000         000  NNNNNN      NNN        TTT        000         000  RRR         RRR
MMM         MMM  000         000  NNN   NNN   NNN        TTT        000         000  RRRRRRRRRRRR
MMM         MMM  000         000  NNN   NNN   NNN        TTT        000         000  RRRRRRRRRRRR
MMM         MMM  000         000  NNN   NNN   NNN        TTT        000         000  RRRRRRRRRRRR
MMM         MMM  000         000  NNN      NNNNNN        TTT        000         000  RRR RRR
MMM         MMM  000         000  NNN      NNNNNN        TTT        000         000  RRR RRR
MMM         MMM  000         000  NNN      NNNNNN        TTT        000         000  RRR RRR
MMM         MMM  000         000  NNN         NNN        TTT        000         000  RRR      RRR
MMM         MMM  000         000  NNN         NNN        TTT        000         000  RRR      RRR
MMM         MMM  000         000  NNN         NNN        TTT        000         000  RRR      RRR
MMM         MMM     000000000     NNN         NNN        TTT           000000000     RRR         RRR
MMM         MMM     000000000     NNN         NNN        TTT           000000000     RRR         RRR
MMM         MMM     000000000     NNN         NNN        TTT           000000000     RRR         RRR
```

```
**FILE**ID**TEMPLATE                              E  4

TTTTTTTTTT  EEEEEEEEEE  MM      MM  PPPPPPPP    LL            AAAAAA    TTTTTTTTTT  EEEEEEEEEE
TTTTTTTTTT  EEEEEEEEEE  MM      MM  PPPPPPPP    LL            AAAAAA    TTTTTTTTTT  EEEEEEEEEE
    TT      EE          MMMM  MMMM  PP      PP  LL          AA      AA      TT      EE
    TT      EE          MMMM  MMMM  PP      PP  LL          AA      AA      TT      EE
    TT      EE          MM  MM  MM  PP      PP  LL          AA      AA      TT      EE
    TT      EE          MM  MM  MM  PP      PP  LL          AA      AA      TT      EE
    TT      EEEEEEEE    MM      MM  PPPPPPPP    LL          AA      AA      TT      EEEEEEEE
    TT      EEEEEEEE    MM      MM  PPPPPPPP    LL          AA      AA      TT      EEEEEEEE
    TT      EE          MM      MM  PP          LL          AAAAAAAAAA      TT      EE
    TT      EE          MM      MM  PP          LL          AAAAAAAAAA      TT      EE
    TT      EE          MM      MM  PP          LL          AA      AA      TT      EE              ....
    TT      EE          MM      MM  PP          LL          AA      AA      TT      EE              ....
    TT      EEEEEEEEEE  MM      MM  PP          LLLLLLLLLL  AA      AA      TT      EEEEEEEEEE      ....
    TT      EEEEEEEEEE  MM      MM  PP          LLLLLLLLLL  AA      AA      TT      EEEEEEEEEE      ....

LL              IIIIII        SSSSSSSS
LL              IIIIII        SSSSSSSS
LL                II        SS
LL                II        SS
LL                II        SS
LL                II        SS
LL                II          SSSSSS
LL                II          SSSSSS
LL                II                SS
LL                II                SS
LL                II                SS
LL                II                SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS
```

```
TEM
VO
;
```

```
0 MODULE TEMPLATE (
0                 IDENT = 'V04-000',
0                 ADDRESSING_MODE(EXTERNAL=GENERAL,
0                                 NONEXTERNAL=LONG_RELATIVE)
0                 ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY: VAX/VMS MONITOR Utility
1 !
1 ! ABSTRACT:
1 !
1 !       The TEMPLATE module contains the routines to create
1 !       templates for the various display screens.
1 !
1 ! ENVIRONMENT:
1 !
1 !       Unprivileged, user mode.
1 !
1 ! AUTHOR: Henry M. Levy , CREATION DATE: 28-April-1977
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-012 TLC1072         Thomas L. Cafarella     17-Apr-1984     11:00
1 !               Add volume name to DISK display.
1 !
1 !       V03-011 TLC1066         Thomas L. Cafarella     01-Apr-1984     11:00
1 !               Add SYSTEM class.
1 !
1 !       V03-010 TLC1060         Thomas L. Cafarella     12-Mar-1984     11:00
1 !               Make multi-file summary work for homogeneous classes.
1 !
1 !       V03-009 TLC1054         Thomas L. Cafarella     07-Mar-1984     11:00
```

```
1 !                   Fix positioning of data lines for homogeneous classes.
1 !
1 !           V03-008 PRS1006         Paul R. Senn            17-FEB-1984     14:00
1 !                   Add support for "computed" items
1 !
1 !           V03-008 TLC1052         Thomas L. Cafarella     17-Feb-1984     11:00
1 !                   Add multi-file summary capability.
1 !
1 !           V03-007 PRS1005         Paul R. Senn            13-JAN-1983     10:00
1 !                   Allow flexible spacing between screen items
1 !
1 !           V03-006 SPC0006         Stephen P. Carney       01-Jul-1983     09:00
1 !                   Change some RWxxx (resource wait state) codes.
1 !
1 !           V03-005 TLC1035         Thomas L. Cafarella     06-Jun-1983     15:00
1 !                   Add homogeneous class type and DISK class.
1 !
1 !           V03-004 TLC1028         Thomas L. Cafarella     14-Apr-1983     16:00
1 !                   Add interactive user interface.
1 !
1 !           V03-004 SPC0001         Stephen P. Carney       25-Mar-1983     15:00
1 !                   Add RWxxx and MUTEX states in place of MWAIT state.
1 !
1 !           V03-003 TLC1020         Thomas L. Cafarella     1-Jul-1982      15:00
1 !                   Remove semi-colon to eliminate BLISS INFO message.
1 !
1 !           V03-002 TLC1010         Thomas L. Cafarella     29-Mar-1982     15:00
1 !                   Eliminate lower-case "a" strings from summary bar graphs.
1 !
1 !           V03-001 TLC1005         Thomas L. Cafarella     25-Mar-1982     17:00
1 !                   Alter vertical spacing for classes with 13 items.
1 !
1 !--
```

```
  1
  1 !
  1 ! TABLE OF CONTENTS:
  1 !
  1
  1 FORWARD ROUTINE
  1         OUTPUT ,                      ! output a counted string to the SCRPKG
  1         POSITION ,                    ! call SCRPKG to position cursor
  1         TEMPLATE ;                    ! build and output display templates
  1
  1
  1 !
  1 ! INCLUDE FILES:
  1 !
  1
  1 LIBRARY 'SYS$LIBRARY:LIB.L32';       ! system service macros and user definitions
  1 REQUIRE 'MONDEFREQ';                 ! private MONITOR control block definitions
  1 REQUIRE 'DSPDEFREQ';                 ! item numbers defined here
  1
  1 BUILTIN EMUL ;                       ! define EMUL VAX hardware function
  1
  1 !
  1 ! COMPILE TIME VARIABLES
  1 !
  1
  1 COMPILETIME
  1     RWAIT_COUNT   = 0 ,              ! counter for the number of RWAITs being defined
  1     RWAIT_DEFINED = RSN$_MAX ;       ! number of RSN$_* wait codes defined in LIB.L32
  1
  1 !
  1 ! MACROS:
  1 !
  1
  1 MACRO
  1
  1 !
  1 ! Counted ascii string macros
  1 !
  1
M 1 CSTRING[] = (UPLIT BYTE(%CHARCOUNT(%STRING(%REMAINING)),
  1                 %STRING(%REMAINING)) )% ,
  1
  1 !       The RWAIT_CSTRING macro is the CSTRING macro plus a counter to
  1 !       keep track of times it was called (how many RWAITS have been defined)
  1
M 1 RWAIT_CSTRING[] = %ASSIGN(RWAIT_COUNT,RWAIT_COUNT+1)
M 1                 (UPLIT BYTE(%CHARCOUNT(%STRING(%REMAINING)),
  1                 %STRING(%REMAINING)) )% ;
  1
  1 !
  1 ! EQUATED SYMBOLS:
  1 !
  1
  1 LITERAL
  1
  1         BELL = 7 ,
  1         ESC = 27 ,
        ALTSET = ('F' ^ 8) + ESC,           ! alternate graphics set
        CR = 13 ,                           ! carriage return
        CURSOR = ('Y' ^ 8) + ESC,           ! position cursor command
        ERASE = ('J' ^ 8) + ESC,            ! erase entire screen
        ERASEEOL = ('K' ^ 8) + ESC,         ! erase to end of line
        FALSE = 0 ,
        HOME = ('H' ^ 8) + ESC,             ! return cursor to top
        LF = 10 ,                           ! line feed
        TRUE = 1 ;

GLOBAL LITERAL

        REGSET = ('G' ^ 8) + ESC ;          ! normal graphics set


!
! OWN STORAGE:
!

OWN
        TOPSTR10: VECTOR[45,BYTE]
                  INITIAL (BYTE(44),BYTE(' [!30W,!30W]  !16AC!AC!5<!#UL!>!AC'),
                           BYTE(ESC),BYTE('F!#*a'),BYTE(ESC),BYTE('G'),BYTE(ESC),BYTE('K')) ;

!
! Table of bit vectors which "illustrate" the pattern of data line
! spacing within the data portion of the display screen. There is
! one bit vector for each possible number of data items (24). Each
! bit vector contains 24 bits representing the lines in the data
! portion of the display screen. A "1" bit means this is a data line;
! a "0" bit means this is a space. The bits read from right to left;
! so, for example, the bit representing line 1 is the right-most.
!


OWN
        SCR_PATTERN:    VECTOR[24,LONG] INITIAL (

                LONG(%B'000000000100000000000000'),     ! 1 data item
                LONG(%B'000000001010000000000000'),     ! 2 data items
                LONG(%B'000000100100100000000000'),     ! 3 data items
                LONG(%B'000000101010100000000000'),     ! 4 data items
                LONG(%B'000010101010100000000000'),     ! 5 data items
                LONG(%B'000010100101001010000000'),     ! 6 data items
                LONG(%B'000010101010101010000000'),     ! 7 data items
                LONG(%B'001010101010101010000000'),     ! 8 data items
                LONG(%B'000011100111001110000000'),     ! 9 data items
                LONG(%B'000110110110110110000000'),     ! 10 data items
                LONG(%B'000110111011101110000000'),     ! 11 data items
                LONG(%B'001110111011101110000000'),     ! 12 data items
                LONG(%B'001111011111011110000000'),     ! 13 data items
                LONG(%B'001111111011111110000000'),     ! 14 data items
                LONG(%B'001111111111111110000000'),     ! 15 data items
                LONG(0),                                ! 16 data items
                LONG(0),                                ! 17 data items
                LONG(0),                                ! 18 data items
                LONG(0),                                ! 19 data items
```

```
                        LONG(0),                                        ! 20 data items
                        LONG(0),                                        ! 21 data items
                        LONG(0),                                        ! 22 data items
                        LONG(0),                                        ! 23 data items
                        LONG(0) );                                      ! 24 data items

!
! One of the above longword elements is moved to the 24-bit vector
! below, based on the number of items in the display. The bit vector
! is then used to determine whether a line in the data portion of the
! screen is to be a space (0) or is to contain data (1).
!

OWN
        SCR_DATA_LINE:  BITVECTOR[24];

!
! Messages
!

BIND

TABSTR = CSTRING('  !7UL.!2ZL !7UL.!2ZL !7UL.!2ZL !7UL.!2ZL'),
TABSTR_PC = CSTRING('   !7UL.!1ZL  !7UL.!1ZL  !7UL.!1ZL  !7UL.!1ZL'),
COUNTSTR = UPLIT BYTE ('!7<!#UL!>'),
CRSTR = CSTRING(%CHAR(CR)),
CLRSTR = CSTRING(%CHAR(ESC),'H',%CHAR(ESC),'J'),
DELSTR = CSTRING(%CHAR(ESC),'J'),
GRAPHICS_ON = CSTRING( %CHAR(ESC) , '1' ),
GRAPHICS_OFF = CSTRING( %CHAR(ESC) , '2' ) ,
HOMESTR = CSTRING(%CHAR(ESC),'H'),

LFSTR = CSTRING(%CHAR(LF)),
NLSTR = CSTRING(%CHAR(CR),%CHAR(LF)),
REPTSTR = UPLIT BYTE('!#*'),
SETVT55 = CSTRING( %CHAR(ESC) , '1' , 'A' , %CHAR(%O'77') , 'I' ,
        %CHAR(%O'57') , %CHAR(ESC) , '2' ),
TOPSTR20 = CSTRING(%CHAR(ESC),'K'),
VHSTSTR20 = CSTRING( '!UL' );

!
! Table of counted strings for Process States
!

GLOBAL BIND

STATELIST = UPLIT (      CSTRING('BAD')    ,
                         CSTRING('COLPG')  ,
                         CSTRING('MWAIT')  ,
                         CSTRING('CEF')    ,
                         CSTRING('PFW')    ,
                         CSTRING('LEF')    ,
                         CSTRING('LEFO')   ,
                         CSTRING('HIB')    ,
                         CSTRING('HIBO')   ,
                         CSTRING('SUSP')   ,
                         CSTRING('SUSPO')  ,
                            CSTRING('FPG')    ,
                            CSTRING('COM')    ,
                            CSTRING('COMO')   ,
                            CSTRING('CUR')
                            ),

RWAITLIST = UPLIT (         RWAIT_CSTRING('RWUDF') ,
                            RWAIT_CSTRING('RWAST') ,
                            RWAIT_CSTRING('RWMBX') ,
                            RWAIT_CSTRING('RWNPG') ,
                            RWAIT_CSTRING('RWPGF') ,
                            RWAIT_CSTRING('RWPAG') ,
                            RWAIT_CSTRING('RWBRK') ,
                            RWAIT_CSTRING('RWIMG') ,
                            RWAIT_CSTRING('RWQUO') ,
                            RWAIT_CSTRING('RWLCK') ,
                            RWAIT_CSTRING('RWSWP') ,
                            RWAIT_CSTRING('RWMPE') ,
                            RWAIT_CSTRING('RWMPB') ,
                            RWAIT_CSTRING('RWSCS') ,
                            RWAIT_CSTRING('RWCLU')
                            ),

! Make sure MONITOR knows all RSN$_* wait states currently defined in LIB.L32

        $ASSUME (RWAIT_COUNT, EQL, RWAIT_DEFINED)

MWAITLIST = UPLIT (         CSTRING('MUTEX')
                            );

```

```
!
! EXTERNAL REFERENCES:
!

EXTERNAL
        MRBPTR ,                        ! address of MRB
        NAME_COL: BYTE ,                ! column number for name string
        BARCHAR: BYTE ,                 ! character to repeat to form bar graphs
        DISPLAYING: BYTE,               ! low bit set => display is active
        FAOSTK: VECTOR[,LONG] ,         ! fao parameter space
        MFSUMSTR ,                      ! fao string segment for control string
        NAMESTR ,                       ! fao string for output of long names
        NORMAL ,                        ! MONITOR normal return status
        PERFTABLE: VECTOR[,BYTE] ,      ! list of performance item descriptors
        ITMSTR_SYS_ALL: BYTE ,          ! item string for SYSTEM /ALL
        SCH$GL_MAXPIX: ADDRESSING_MODE(LONG_RELATIVE) , ! max process index
        SCH$GL_PCBVEC: ADDRESSING_MODE(LONG_RELATIVE) , ! address of PCB pointer list
        VT55XINCR ;                     ! incr from bar to bar

EXTERNAL LITERAL
        FAOCTR_SIZE ,                   ! size of FAO control string
        FIRST_DATA_LINE,                ! line number of first data line on screen
        LAST_DATA_LINE,                 ! line number of last data line on screen
        VTDATALINES,                    ! number of data lines on the screen
        NAME_COL_TAB,                   ! starting column of names -- tabular display
        NAME_COL_BAR,                   ! starting column of names -- bar graph
        NAME_COL_MFSUM,                 ! starting column of names -- multi-file summary
        MAX_NAME_SIZE,                  ! max size of name (label) string
        WIDE_NAME_SIZE,                 ! size of name (label) string for a wide display (DISK)
        ECOUNT_SYS_ALL,                 ! no. of elements for SYSTEM /ALL
        MAXBARS ,                       ! max characters on horizontal histogram
        VT55CWIDTH,                     ! max characters on bottom axis
        VTHEIGHT,                       ! height of screen
        VTWIDTH ;                       ! width of screen

EXTERNAL ROUTINE
        PUT_TO_SCREEN ,                 ! rtn to xlate & annex a string to SYS$OUTPUT buffer
        LIB$GET_VM ,                    ! rtn to acquire virtual memory
        SCR$SET_CURSOR ;                ! rtn to annex a cursor positioning esc seq to SYS$OUTPUT
```

```
1 GLOBAL ROUTINE TEMPLATE( DCDB ) =
2 BEGIN
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine formats and displays the name strings for tabular
2 !       and bar graph displays of current, average, min and max values.
2 !       It also builds the FAO control string for the actual data on the
2 !       first call per class.
2 !
2 ! INPUTS:
2 !
2 !       DCDB   - address of class descriptor block for class being displayed.
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       PERFTABLE - address of table of contiguous IDB's.
2 !
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       Name string for each item in the display for this class sent
2 !       directly to screen package (via call to PUT_TO_SCREEN).
2 !
2 !       On first call to this routine for this class, a buffer is
2 !       obtained for the FAO control string to output the data values.
2 !       It is filled with the necessary FAO control information and
2 !       its address and length are stored in the CDB$A_FAOCTR and
2 !       CDB$L_FAOCTR fields, respectively.
2 !
2 ! ROUTINE VALUE:
2 !
2 !       NORMAL, or possible failing status from LIB$GET_VM.
2 !
2 ! SIDE EFFECTS:
2 !
2 !       none
2 !--
2
2 LOCAL
2       I,                                      ! data item index
2       ITEMS,                                  ! count of data items
2       ITMSTR,                                 ! pointer to first item token
2       POINTER,                                ! pointer into fao control string buffer
2       STATUS,                                 ! return status
2       XPOS,                                   ! column address
2       YPOS,                                   ! row address
2       ROW_OFFSET;                             ! constant added to row number for m.f. summary
2 MAP
2       DCDB:   REF BLOCK[,BYTE] ,              ! address CDB structure
2       MRBPTR: REF BLOCK[,BYTE] ,              ! address MRB structure
```

```
;  392          1524  2         ITMSTR: REF VECTOR[,BYTE] ;               ! item byte string
```

```
2 IF .MRBPTR[MRB$V_MFSUM]                         ! if this is a multi-file summary
2     THEN  ROW_OFFSET = 2                        !   then display the data rows lower
2     ELSE  ROW_OFFSET = 0 ;                      !   else do not offset
2
2 IF .DCDB[CDB$V_HOMOG]                           ! if this is a homogeneous class,
2     THEN  ITEMS = VTDATALINES                   !   always use the whole screen,
2     ELSE  ITEMS = .DCDB[CDB$L_ECOUNT] ;         !   else get just no. of elts to display
2
2 IF .DCDB[CDB$V_SYSCLS]                          ! if this is the SYSTEM class,
2     THEN ITEMS = ECOUNT_SYS_ALL ;               !   get a special ECOUNT
2
2 SCR_DATA_LINE = 0;                              ! zero out display bit string
2
2 !
2 ! Set up bit string controlling spacing.
2 ! The CDB display control string is only a word in length, rather than 24 bits.
2 ! This is to save space, since only 15 of the 24 bits in the default bit
2 ! strings are actually used.
2 !
2
2 IF .DCDB[CDB$W_DISPCTL] EQL 0                   ! if display control is 0
2     THEN  SCR_DATA_LINE = .(SCR_PATTERN[.ITEMS-1])<0,24>    ! use default spacing
2     ELSE  SCR_DATA_LINE<7,15> = .(DCDB[CDB$W_DISPCTL])<0,15> ; ! else use spacing specified in CDB
2 !
2 ! Output name string for each item in this heterogeneous class
2 !
2
2 IF .MRBPTR[MRB$V_MFSUM] OR .DCDB[CDB$V_WIDE]    ! if this is a multi-file summary or a wide screen
2     THEN  NAME_COL = NAME_COL_MFSUM             !   start the names here
2     ELSE  IF .DCDB[CDB$B_ST] EQL ALL_STAT       ! if this is a tabular display,
2             THEN  NAME_COL = NAME_COL_TAB       !   start the names here
2             ELSE  NAME_COL = NAME_COL_BAR ;     !   else start there for bar graph
2
2 IF NOT .DCDB[CDB$V_HOMOG]                       ! if this is a heterogeneous class,
2 THEN
3   BEGIN
3
3   I = 0 ;                                       ! initialize data item index
3   ITMSTR = .DCDB[CDB$A_ITMSTR] ;                ! get address of item byte string
3
3   IF .DCDB[CDB$V_SYSCLS] AND .DCDB[CDB$B_ST] EQL ALL_STAT      ! if this is the SYSTEM tabular display,
3     THEN ITMSTR = ITMSTR_SYS_ALL ;                             !   get a special ITMSTR
3
3   INCR YPOS FROM FIRST_DATA_LINE TO LAST_DATA_LINE ! loop once for each line in
3   DO                                            ! ... data portion of screen
4     BEGIN
4
4     !
4     ! Find the IDB for this item.  Output the long name
4     ! string, preceded by the correct cursor positioning
4     ! sequence to space them out evenly.
4     !
4
4     LOCAL
4         DIDB: REF BLOCK[,BYTE] ,
4         NAME ,
4         NEXT ;
```

```
4         IF .SCR_DATA_LINE[.YPOS-1]                                 ! if this is a data line,
4         THEN
5             BEGIN
5             NEXT = .ITMSTR[.I] ;                                   ! get next token
5             DIDB = PERFTABLE[ .NEXT * IDB$K_ILENGTH ] ;            ! addr of IDB
5             NAME = .DIDB[IDB$A_LNAME] ;                            ! address of name string
5             POSITION( .YPOS + .ROW_OFFSET , .NAME_COL ) ;          ! position to this item
5             OUTPUT( .NAME ) ;                                      ! output name string
5             IF .DIDB[IDB$V_PCNT] EQL 1                             ! if this is a pcnt item
5             THEN I = .I + 2                                        ! move past item used for calc
5             ELSE I = .I + 1;                                       ! point index to next data item
4             END;
4
3         END;
2     END;
```

```
2  !
2  ! If this is the first time thru for this class,
2  ! obtain and build the FAO control string to insert
2  ! the data values for the items at data display time.
2  !
2
2  IF .DCDB[CDB$A_FAOCTR] EQL 0  OR  NOT .DISPLAYING              ! if no fao control string yet
2  THEN                                                           ! ... OR in summary processing
3      BEGIN
3      LOCAL
3          FAOCSIZE ;                                             ! holds faoctr size
3      IF .DCDB[CDB$A_FAOCTR] EQL 0                               ! if no control string buffer yet,
3      THEN
4          BEGIN
4          FAOCSIZE = FAOCTR_SIZE ;                               ! initialize its size
4          STATUS = LIB$GET_VM(FAOCSIZE,DCDB[CDB$A_FAOCTR]);      ! get the memory for it
4          IF NOT .STATUS THEN RETURN .STATUS ;                   ! return if error
3          END;
3
3      POINTER = .DCDB[CDB$A_FAOCTR] ;                            ! start pointer at beg of FAO buffer
3
3      IF .DCDB[CDB$B_ST] EQL ALL_STAT OR .MRBPTR[MRB$V_MFSUM]    ! if this is a tabular display,
3      THEN                                                       !   set up control string accordingly
4          BEGIN
4          LOCAL
4              COL_OFFSET,                                        ! holds offset from usual column where data
4              CUR_TABSTR ;                                       ! holds addr of FAO control string segment
4          IF .DCDB[CDB$V_WIDE]                                   ! if a wide-screen display,
4              THEN COL_OFFSET = WIDE_NAME_SIZE                   !   then set a wide offset
4              ELSE COL_OFFSET = MAX_NAME_SIZE ;                  !   otherwise, take the usual width
4          XPOS = .NAME_COL + .COL_OFFSET ;                       ! starting column
4          DCDB[CDB$B_FAOPRELEN] = 0 ;                            ! length of FAO prefix
4
4          IF .MRBPTR[MRB$V_MFSUM]                                ! if this is a multi-file summary,
4              THEN  CUR_TABSTR = MFSUMSTR                        !   get the appropriate FAO control str segm
4              ELSE IF .DCDB[CDB$V_PERCENT]                       ! if this is a percent display,
4                        THEN  CUR_TABSTR = TABSTR_PC             !   get the appropriate FAO control str segm
4                        ELSE  CUR_TABSTR = TABSTR;               !   else get the other one
4
4          INCR YPOS FROM FIRST_DATA_LINE TO LAST_DATA_LINE       ! loop once for each line in
4          DO                                                     ! ... data portion of screen
5              BEGIN
5              IF .SCR_DATA_LINE[.YPOS-1]                         ! if this is a data line,
5              THEN
6                  BEGIN
6                  (.POINTER)<0,16> = CURSOR ;                    ! insert position command
6                  ( POINTER = .POINTER + 2 )<0,8> = .YPOS + .ROW_OFFSET ; ! insert row number
6                  ( POINTER = .POINTER + 1 )<0,8> = .XPOS ;      ! insert column number
6                  POINTER = .POINTER + 1 ;                       ! update to skip last inserted byte
6                  CH$MOVE( .(.CUR_TABSTR)<0,8> , (.CUR_TABSTR)+1 , .POINTER ) ; ! move conversion control stri
6                  POINTER = .POINTER + .(.CUR_TABSTR)<0,8> ;     ! update pointer
6                  IF .YPOS EQL FIRST_DATA_LINE                   ! if first time thru the loop,
6                      THEN DCDB[CDB$B_FAOSEGLEN] = .POINTER - .DCDB[CDB$A_FAOCTR] - .DCDB[CDB$B_FAOPRELEN] ;
6                                                                 ! compute length of a single segment
5                  END;
4              END;
4
```

```
;  524          1654  4         END
```

```
3       ELSE                                                    ! bar graph display -- set up ctrl string for it
4           BEGIN
4
4           ! Now build the fao control string to output a bar graph
4           ! at run time.  The control string contains for each line:
4           !       position row and column to left of grid
4           !       write count
4           !       re-position row and column inside grid
4           !       output 'n' bar characters
4           !       delete to end of line
4           !
4           LOCAL
4               XPOSBAR ,                                       ! column number of beg of bar
4               XPOSCOUNT ;                                     ! column number of count field
4
4           XPOSCOUNT = 30 ;                                    ! starting column of count field
4           XPOSBAR = 39 ;                                      ! starting column of bar field
4           (.POINTER) <0,16> = ALTSET ;                        ! start filling ctrl string (alternate graphics)
4           POINTER = .POINTER + 2 ;                            ! skip to next position
4           DCDB[CDB$B_FAOPRELEN] = 2 ;                         ! ... and store length of FAO prefix
4
4           INCR YPOS FROM FIRST_DATA_LINE TO LAST_DATA_LINE ! loop once for each line in
4           DO                                                  ! ... data portion of screen
5               BEGIN
5               IF .SCR_DATA_LINE[.YPOS-1]                              ! if this is a data line,
5               THEN
6                   BEGIN
6                   (.POINTER)<0,16> = CURSOR ;                         ! insert position command
6                   (POINTER = .POINTER + 2 )<0,8> = .YPOS ;            ! next Y position
6                   (POINTER = .POINTER + 1 )<0,8> = .XPOSCOUNT ;       ! X position for count
6                   POINTER = .POINTER + 1 ;                            ! next buffer position
6                   CH$MOVE( 9 , COUNTSTR , .POINTER ) ;                ! move count directive
6                   (POINTER = .POINTER+9)<0,16> = CURSOR ;             ! insert control to position to
6                   (POINTER = .POINTER+2)<0,8> = .YPOS ;               ! stay in same row
6                   (POINTER = .POINTER+1)<0,8> = .XPOSBAR ;            ! column for bar field
6                   POINTER = .POINTER + 1 ;                            ! next buffer position
6                   CH$MOVE( 3 , REPTSTR , .POINTER ) ;                 ! move repeat control
6                   (POINTER = .POINTER + 3)<0,8> = .BARCHAR ;          ! insert literal character to use for graph
6                   (POINTER = .POINTER+1)<0,16> = ERASEEOL ;           ! delete rest of line
6                   POINTER = .POINTER + 2 ;                            ! next buffer position
6                   IF .YPOS EQL FIRST_DATA_LINE                        ! if first time thru the loop,
6                       THEN DCDB[CDB$B_FAOSEGLEN] = .POINTER - .DCDB[CDB$A_FAOCTR] - .DCDB[CDB$B_FAOPRELEN] ;
6                                                                       ! compute length of a single segment
5                   END;
4               END;
4
4           (.POINTER)<0,16> = REGSET ;                         ! restore normal char set
4           POINTER = .POINTER + 2 ;                            ! update position
3           END;
3
3       !
3       ! Insert length of created string into CDB
3       !
3
3       DCDB[CDB$L_FAOCTR] = .POINTER - .DCDB[CDB$A_FAOCTR] ;
2       END ;
2 RETURN .NORMAL ;                                              ! return with no errors
```

```
;  583          1712  1 END;

                                                                                                 .TITLE  TEMPLATE
                                                                                                 .IDENT  \V04-000\

                                                                                                 .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                                           29   00000 P.AAA:  .BYTE   41
55   37   21   20   4C   5A   32   21   2E   4C   55   37   21   20   20   00001         .ASCII  \  !7UL.!2ZL !7UL.!2ZL !7UL.!2ZL !7UL.!2Z\
5A   32   21   2E   4C   55   37   21   20   4C   5A   32   21   2E   4C   00010
                         5A   32   21   2E   4C   55   37   21   20   4C   0001F
                                                                      4C   00029         .ASCII  \L\
                                                                      20   0002A P.AAB:  .BYTE   45
21   20   20   4C   5A   31   21   2E   4C   55   37   21   20   20   20   0002B         .ASCII  \   !7UL.!1ZL  !7UL.!1ZL  !7UL.!1ZL  !7UL\
2E   4C   55   37   21   20   20   4C   5A   31   21   2E   4C   55   37   0003A
                         4C   55   37   21   20   20   4C   5A   31   21   00049
                                                  4C   5A   31   21   2E   00053         .ASCII  \.!1ZL\
                              3E   21   4C   55   23   21   3C   37   21   00058 P.AAC:  .ASCII  \!7<!#UL!>\
                                                                      01   00061 P.AAD:  .BYTE   1
                                                                      0D   00062         .ASCII  <13>
                                                                      04   00063 P.AAE:  .BYTE   4
                                                       4A   1B   48   1B   00064         .ASCII  <27>\H\<27>\J\
                                                                      02   00068 P.AAF:  .BYTE   2
                                                                 4A   1B   00069         .ASCII  <27>\J\
                                                                      02   0006B P.AAG:  .BYTE   2
                                                                 31   1B   0006C         .ASCII  <27>\1\
                                                                      02   0006E P.AAH:  .BYTE   2
                                                                 32   1B   0006F         .ASCII  <27>\2\
                                                                      02   00071 P.AAI:  .BYTE   2
                                                                 48   1B   00072         .ASCII  <27>\H\
                                                                      01   00074 P.AAJ:  .BYTE   1
                                                                      0A   00075         .ASCII  <10>
                                                                      02   00076 P.AAK:  .BYTE   2
                                                                 0A   0D   00077         .ASCII  <13><10>
                                                            2A   23   21   00079 P.AAL:  .ASCII  \!#*\
                                                                      08   0007C P.AAM:  .BYTE   8
                                   32   1B   2F   49   3F   41   31   1B   0007D         .ASCII  <27>\1A?I/\<27>\2\
                                                                      02   00085 P.AAN:  .BYTE   2
                                                                 4B   1B   00086         .ASCII  <27>\K\
                                                                      03   00088 P.AAO:  .BYTE   3
                                                            4C   55   21   00089         .ASCII  \!UL\
                                                                      03   0008C P.AAQ:  .BYTE   3
                                                            44   41   42   0008D         .ASCII  \BAD\
                                                                      05   00090 P.AAR:  .BYTE   5
                                                  47   50   4C   4F   43   00091         .ASCII  \COLPG\
                                                                      05   00096 P.AAS:  .BYTE   5
                                                  54   49   41   57   4D   00097         .ASCII  \MWAIT\
                                                                      03   0009C P.AAT:  .BYTE   3
                                                            46   45   43   0009D         .ASCII  \CEF\
                                                                      03   000A0 P.AAU:  .BYTE   3
                                                            57   46   50   000A1         .ASCII  \PFW\
                                                                      03   000A4 P.AAV:  .BYTE   3
                                                            46   45   4C   000A5         .ASCII  \LEF\
                                                                      04   000A8 P.AAW:  .BYTE   4
                                                       4F   46   45   4C   000A9         .ASCII  \LEFO\
                                                                      03   000AD P.AAX:  .BYTE   3
```

```
                                         42 49 48  000AE         .ASCII   \HIB\                                          ;
                                               04  000B1 P.AAY:  .BYTE    4                                              ;
                                      4F 42 49 48  000B2         .ASCII   \HIBO\                                         ;
                                               04  000B6 P.AAZ:  .BYTE    4                                              ;
                                      50 53 55 53  000B7         .ASCII   \SUSP\                                         ;
                                               05  000BB P.ABA:  .BYTE    5                                              ;
                                   4F 50 53 55 53  000BC         .ASCII   \SUSPO\                                        ;
                                               03  000C1 P.ABB:  .BYTE    3                                              ;
                                         47 50 46  000C2         .ASCII   \FPG\                                          ;
                                               03  000C5 P.ABC:  .BYTE    3                                              ;
                                         4D 4F 43  000C6         .ASCII   \COM\                                          ;
                                               04  000C9 P.ABD:  .BYTE    4                                              ;
                                      4F 4D 4F 43  000CA         .ASCII   \COMO\                                         ;
                                               03  000CE P.ABE:  .BYTE    3                                              ;
                                         52 55 43  000CF         .ASCII   \CUR\                                          ;
                                                   000D2         .BLKB    2                                              ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  000D4 P.AAP:  .ADDRESS P.AAQ, P.AAR, P.AAS, P.AAT, P.AAU, -  ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  000EC                  P.AAV, P.AAW, P.AAX, P.AAY, P.AAZ, P.ABA, - ;
                              00000000' 00000000' 00000000'  00104                  P.ABB, P.ABC, P.ABD, P.ABE          ;
                                               05  00110 P.ABG:  .BYTE    5                                              ;
                                   46 44 55 57 52  00111         .ASCII   \RWUDF\                                        ;
                                               05  00116 P.ABH:  .BYTE    5                                              ;
                                   54 53 41 57 52  00117         .ASCII   \RWAST\                                        ;
                                               05  0011C P.ABI:  .BYTE    5                                              ;
                                   58 42 4D 57 52  0011D         .ASCII   \RWMBX\                                        ;
                                               05  00122 P.ABJ:  .BYTE    5                                              ;
                                   47 50 4E 57 52  00123         .ASCII   \RWNPG\                                        ;
                                               05  00128 P.ABK:  .BYTE    5                                              ;
                                   46 47 50 57 52  00129         .ASCII   \RWPGF\                                        ;
                                               05  0012E P.ABL:  .BYTE    5                                              ;
                                   47 41 50 57 52  0012F         .ASCII   \RWPAG\                                        ;
                                               05  00134 P.ABM:  .BYTE    5                                              ;
                                   4B 52 42 57 52  00135         .ASCII   \RWBRK\                                        ;
                                               05  0013A P.ABN:  .BYTE    5                                              ;
                                   47 4D 49 57 52  0013B         .ASCII   \RWIMG\                                        ;
                                               05  00140 P.ABO:  .BYTE    5                                              ;
                                   4F 55 51 57 52  00141         .ASCII   \RWQUO\                                        ;
                                               05  00146 P.ABP:  .BYTE    5                                              ;
                                   4B 43 4C 57 52  00147         .ASCII   \RWLCK\                                        ;
                                               05  0014C P.ABQ:  .BYTE    5                                              ;
                                   50 57 53 57 52  0014D         .ASCII   \RWSWP\                                        ;
                                               05  00152 P.ABR:  .BYTE    5                                              ;
                                   45 50 4D 57 52  00153         .ASCII   \RWMPE\                                        ;
                                               05  00158 P.ABS:  .BYTE    5                                              ;
                                   42 50 4D 57 52  00159         .ASCII   \RWMPB\                                        ;
                                               05  0015E P.ABT:  .BYTE    5                                              ;
                                   53 43 53 57 52  0015F         .ASCII   \RWSCS\                                        ;
                                               05  00164 P.ABU:  .BYTE    5                                              ;
                                   55 4C 43 57 52  00165         .ASCII   \RWCLU\                                        ;
                                                   0016A         .BLKB    2                                              ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  0016C P.ABF:  .ADDRESS P.ABG, P.ABH, P.ABI, P.ABJ, P.ABK, -  ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  00184                  P.ABL, P.ABM, P.ABN, P.ABO, P.ABP, P.ABQ, - ;
                              00000000' 00000000' 00000000'  0019C                  P.ABR, P.ABS, P.ABT, P.ABU          ;
                                               05  001A8 P.ABW:  .BYTE    5                                              ;
                                   58 45 54 55 4D  001A9         .ASCII   \MUTEX\                                        ;
                                                   001AE         .BLKB    2                                              ;
                                        00000000'  001B0 P.ABV:  .ADDRESS P.ABW                                          ;
```

```
                                                                    I 5
TEMPLATE                                                            16-Sep-1984 02:18:37    VAX-11 Bliss-32 V4.0-742              Page  17
V04-000                                                             14-Sep-1984 12:45:05    [MONTOR.SRC]TEMPLATE.B32;1                  (7)

                                                                                    .PSECT  $OWN$,NOEXE,2

                                                            2C   00000 TOPSTR10:
                                                                                    .BYTE   44
21  20  20  5D  57  4F  33  21  2C  57  4F  33  21  5B  20   00001                  .ASCII  \ [!30W,!30W]  !16AC!AC!5<!#UL!>!AC\
21  4C  55  23  21  3C  35  21  43  41  21  43  41  36  31   00010
                                            43  41  21  3E   0001F
                                                        1B   00023                  .BYTE   27
                                            61  2A  23  21  46   00024              .ASCII  \F!#*a\
                                                        1B   00029                  .BYTE   27
                                                        47   0002A                  .ASCII  \G\
                                                        1B   0002B                  .BYTE   27
                                                        4B   0002C                  .ASCII  \K\
                                                             0002D                  .BLKB   3
                                                  00004000   00030 SCR_PATTERN:
                                                                                    .LONG   16384
                                                  0000A000   00034                  .LONG   40960
                                                  00024800   00038                  .LONG   149504
                                                  0002A800   0003C                  .LONG   174080
                                                  00055400   00040                  .LONG   349184
                                                  000A5280   00044                  .LONG   676480
                                                  000AAA80   00048                  .LONG   699008
                                                  002AAA80   0004C                  .LONG   2796160
                                                  000E7380   00050                  .LONG   947072
                                                  001B6D80   00054                  .LONG   1797504
                                                  001BBB80   00058                  .LONG   1817472
                                                  003BBB80   0005C                  .LONG   3914624
                                                  003DF780   00060                  .LONG   4061056
                                                  003FBF80   00064                  .LONG   4177792
                                                  003FFF80   00068                  .LONG   4194176
                                                  00000000   0006C                  .LONG   0
                                                  00000000   00070                  .LONG   0
                                                  00000000   00074                  .LONG   0
                                                  00000000   00078                  .LONG   0
                                                  00000000   0007C                  .LONG   0
                                                  00000000   00080                  .LONG   0
                                                  00000000   00084                  .LONG   0
                                                  00000000   00088                  .LONG   0
                                                  00000000   0008C                  .LONG   0
                                                             00090 SCR_DATA_LINE:
                                                                                    .BLKB   3

                                                                   REGSET==             18203
                                                                   TABSTR=              P.AAA
                                                                   TABSTR_PC=           P.AAB
                                                                   COUNTSTR=            P.AAC
                                                                   CRSTR=               P.AAD
                                                                   CLRSTR=              P.AAE
                                                                   DELSTR=              P.AAF
                                                                   GRAPHICS_ON=         P.AAG
                                                                   GRAPHICS_OFF=        P.AAH
                                                                   HOMESTR=             P.AAI
                                                                   LFSTR=               P.AAJ
                                                                   NLSTR=               P.AAK
                                                                   REPTSTR=             P.AAL
                                                                   SETVT55=             P.AAM
```

```
                                                              J  5
TEMPLATE                                                      16-Sep-1984 02:18:37    VAX-11 Bliss-32 V4.0-742              Page  18
V04-000                                                       14-Sep-1984 12:45:05    [MONTOR.SRC]TEMPLATE.B32;1                 (7)

                                                               TOPSTR20=            P.AAN
                                                               VHSTSTR20=           P.AAO
                                                               STATELIST==          P.AAP
                                                               RWAITLIST==          P.ABF
                                                               MWAITLIST==          P.ABV
                                                                      .EXTRN  MRBPTR, NAME_COL
                                                                      .EXTRN  BARCHAR, DISPLAYING
                                                                      .EXTRN  FAOSTK, MFSUMSTR
                                                                      .EXTRN  NAMESTR, NORMAL
                                                                      .EXTRN  PERFTABLE, ITMSTR_SYS_ALL
                                                                      .EXTRN  SCH$GL_MAXPIX, SCH$GL_PCBVEC
                                                                      .EXTRN  VT55XINCR, FAOCTR_SIZE
                                                                      .EXTRN  FIRST_DATA_LINE
                                                                      .EXTRN  LAST_DATA_LINE, VTDATALINES
                                                                      .EXTRN  NAME_COL_TAB, NAME_COL_BAR
                                                                      .EXTRN  NAME_COL_MFSUM, MAX_NAME_SIZE
                                                                      .EXTRN  WIDE_NAME_SIZE, ECOUNT_SYS_ALL
                                                                      .EXTRN  MAXBARS, VT55CWIDTH
                                                                      .EXTRN  VTHEIGHT, VTWIDTH
                                                                      .EXTRN  PUT_TO_SCREEN, LIB$GET_VM
                                                                      .EXTRN  SCR$SET_CURSOR

                                                                      .PSECT  $CODE$,NOWRT,2

                                                  0FFC 00000          .ENTRY  TEMPLATE, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-;  1467
                                                                              R11
                                    5E              08  C2 00002      SUBL2   #8, SP                                   :
                                    51 00000000G    00  D0 00005      MOVL    MRBPTR, R1                               :  1525
                         05     44  A1              03  E1 0000C      BBC     #3, 68(R1), 1$                           :
                                    5B              02  D0 00011      MOVL    #2, ROW_OFFSET                           :  1526
                                                    02  11 00014      BRB     2$                                       :
                                                    5B  D4 00016 1$:  CLRL    ROW_OFFSET                               :  1527
                                    58          04  AC  D0 00018 2$:  MOVL    DCDB, R8                                 :  1529
                         09     4B  A8              05  E1 0001C      BBC     #5, 75(R8), 3$                           :
                                    50 00000000G    8F  D0 00021      MOVL    #VTDATALINES, ITEMS                      :  1530
                                                    04  11 00028      BRB     4$                                       :
                                    50          18  A8  D0 0002A 3$:  MOVL    24(R8), ITEMS                            :  1531
                                    07          4C  A8  E9 0002E 4$:  BLBC    76(R8), 5$                               :  1533
                                    50 00000000G    8F  D0 00032      MOVL    #ECOUNT_SYS_ALL, ITEMS                   :  1534
00000000'  EF            18         00              00  F0 00039 5$:  INSV    #0, #0, #24, SCR_DATA_LINE               :  1536
                                                36  A8  B5 00042      TSTW    54(R8)                                   :  1545
                                                    10  12 00045      BNEQ    6$                                       :
00000000'  EF            18         00 00000000'EF40    F0 00047      INSV    SCR_PATTERN-4[ITEMS], #0, #24, -         :  1546
                                                                              SCR_DATA_LINE                            :
                                                    0A  11 00055      BRB     7$                                       :
00000000'  EF            0F         07          36  A8  F0 00057 6$:  INSV    54(R8), #7, #15, SCR_DATA_LINE           :  1547
                         05     44  A1              03  E0 00061 7$:  BBS     #3, 68(R1), 8$                           :  1552
                         0A     4C  A8              03  E1 00066      BBC     #3, 76(R8), 9$                           :
                           00000000G  00       00G  8F  90 0006B 8$:  MOVB    #NAME_COL_MFSUM, NAME_COL                :  1553
                                                    17  11 00073      BRB     11$                                      :
                                                42  A8  95 00075 9$:  TSTB    66(R8)                                   :  1554
                                                    0A  12 00078      BNEQ    10$                                      :
                           00000000G  00       00G  8F  90 0007A      MOVB    #NAME_COL_TAB, NAME_COL                  :  1555
                                                    08  11 00082      BRB     11$                                      :
                           00000000G  00       00G  8F  90 00084 10$: MOVB    #NAME_COL_BAR, NAME_COL                  :  1556
                         6C     4B  A8              05  E0 0008C 11$: BBS     #5, 75(R8), 16$                          :  1558
                                                    55  D4 00091      CLRL    I                                        :  1562
```

```
                     54          1C   A8   D0 00093          MOVL     28(R8), ITMSTR                                   ; 1563
                     0C          4C   A8   E9 00097          BLBC     76(R8), 12$                                      ; 1565
                                 42   A8   95 0009B          TSTB     66(R8)                                           ;
                                      07   12 0009E          BNEQ     12$                                              ;
                     54 00000000G     00   9E 000A0          MOVAB    ITMSTR_SYS_ALL, ITMSTR                           ; 1566
        52 00000000G 8F               01   C3 000A7  12$:    SUBL3    #1, #FIRST_DATA_LINE, YPOS                       ; 1568
                                      44   11 000AF          BRB      15$                                              ;
                     50          FF   A2   9E 000B1  13$:    MOVAB    -1(R2), R0                                       ; 1582
        38 00000000' EF               50   E1 000B5          BBC      R0, SCR_DATA_LINE, 15$                           ;
                     50             6544   9A 000BD          MOVZBL   (I)[ITMSTR], NEXT                                ; 1585
                     50               11   C4 000C1          MULL2    #17, R0                                          ; 1586
                     53 00000000G0040      9E 000C4          MOVAB    PERFTABLE[R0], DIDB                              ;
                     56          04   A3   D0 000CC          MOVL     4(DIDB), NAME                                    ; 1587
                     7E 00000000G     00   9A 000D0          MOVZBL   NAME_COL, -(SP)                                  ; 1588
                                    6B42   9F 000D7          PUSHAB   (ROW_OFFSET)[YPOS]                               ;
           00000000V EF               02   FB 000DA          CALLS    #2, POSITION                                     ;
                                      56   DD 000E1          PUSHL    NAME                                             ; 1589
           00000000V EF               01   FB 000E3          CALLS    #1, OUTPUT                                       ;
                     05          10   A3   E9 000EA          BLBC     16(DIDB), 14$                                    ; 1590
                     55               02   C0 000EE          ADDL2    #2, I                                            ; 1591
                                      02   11 000F1          BRB      15$                                              ;
                                      55   D6 000F3  14$:    INCL     I                                                ; 1592
        B4           52 00000000G     8F   F3 000F5  15$:    AOBLEQ   #LAST_DATA_LINE, YPOS, 13$                       ; 1568
                     5A          04   A8   9E 000FD  16$:    MOVAB    4(R8), R10                                       ; 1603
                                      50   D4 00101          CLRL     R0                                               ;
                                      6A   D5 00103          TSTL     (R10)                                            ;
                                      04   12 00105          BNEQ     17$                                              ;
                                      50   D6 00107          INCL     R0                                               ;
                                      0A   11 00109          BRB      18$                                              ;
                     03 00000000G     00   E9 0010B  17$:    BLBC     DISPLAYING, 18$                                  ;
                                    014D   31 00112          BRW      32$                                              ;
                     18               50   E9 00115  18$:    BLBC     R0, 19$                                          ; 1608
                04   AE 00000000G     8F   D0 00118          MOVL     #FAOCTR_SIZE, FAOCSIZE                           ; 1611
                                      5A   DD 00120          PUSHL    R10                                              ; 1612
                                 08   AE   9F 00122          PUSHAB   FAOCSIZE                                         ;
           00000000G 00               02   FB 00125          CALLS    #2, LIB$GET_VM                                   ;
                     01               50   E8 0012C          BLBS     STATUS, 19$                                      ; 1613
                                           04 0012F          RET                                                       ;
                     56               6A   D0 00130  19$:    MOVL     (R10), POINTER                                   ; 1616
                                 42   A8   95 00133          TSTB     66(R8)                                           ; 1618
                                      0F   13 00136          BEQL     20$                                              ;
                     50 00000000G     00   D0 00138          MOVL     MRBPTR, R0                                       ;
        03       44  A0               03   E0 0013F          BBS      #3, 68(R0), 20$                                  ;
                                    0099   31 00144          BRW      28$                                              ;
        09       4C  A8               03   E1 00147  20$:    BBC      #3, 76(R8), 21$                                  ; 1624
                     50 00000000G     8F   D0 0014C          MOVL     #WIDE_NAME_SIZE, COL_OFFSET                      ; 1625
                                      07   11 00153          BRB      22$                                              ;
                     50 00000000G     8F   D0 00155  21$:    MOVL     #MAX_NAME_SIZE, COL_OFFSET                       ; 1626
                     51 00000000G     00   9A 0015C  22$:    MOVZBL   NAME_COL, R1                                     ; 1627
        6E           51               50   C1 00163          ADDL3    COL_OFFSET, R1, XPOS                             ;
                                 41   A8   94 00167          CLRB     65(R8)                                           ; 1628
                     50 00000000G     00   D0 0016A          MOVL     MRBPTR, R0                                       ; 1630
        09       44  A0               03   E1 00171          BBC      #3, 68(R0), 23$                                  ;
                     59 00000000G     00   9E 00176          MOVAB    MFSUMSTR, CUR_TABSTR                             ; 1631
                                      14   11 0017D          BRB      25$                                              ;
                     09          45   A8   E9 0017F  23$:    BLBC     69(R8), 24$                                      ; 1632
                     59 00000000'     EF   9E 00183          MOVAB    TABSTR_PC, CUR_TABSTR                            ; 1633
```

```
                                             07  11 0018A      BRB      25$                                        ;
                           59 00000000'      EF  9E 0018C 24$: MOVAB    TABSTR, CUR_TABSTR                         ; 1634
              57 00000000G 8F                01  C3 00193 25$: SUBL3    #1, #FIRST_DATA_LINE, YPOS                 ; 1636
                                             39  11 0019B      BRB      27$                                        ;
                           50        FF      A7  9E 0019D 26$: MOVAB    -1(R7), R0                                 ; 1639
              2D 00000000' EF                50  E1 001A1      BBC      R0, SCR_DATA_LINE, 27$                     ;
                           86      591B      8F  B0 001A9      MOVW     #22811, (POINTER)+                         ; 1642
              86           57                5B  81 001AE      ADDB3    ROW_OFFSET, YPOS, (POINTER)+               ; 16-3
                           86                6E  90 001B2      MOVB     XPOS, (POINTER)+                           ; 1644
                           50                69  9A 001B5      MOVZBL   (CUR_TABSTR), R0                           ; 1646
              66        01 A9                50  28 001B8      MOVC3    R0, 1(CUR_TABSTR), (POINTER)               ;
                           50                69  9A 001BD      MOVZBL   (CUR_TABSTR), R0                           ; 1647
                           56                50  C0 001C0      ADDL2    R0, POINTER                                ;
                 00000000G 8F                57  D1 001C3      CMPL     YPOS, #FIRST_DATA_LINE                     ; 1648
                                             0A  12 001CA      BNEQ     27$                                        ;
              50           56                6A  C3 001CC      SUBL3    (R10), POINTER, R0                         ; 1649
         40   A8           50        41      A8  83 001D0      SUBB3    65(R8), R0, 64(R8)                         ;
              BF           57 00000000G      8F  F3 001D6 27$: AOBLEQ   #LAST_DATA_LINE, YPOS, 26$                 ; 1636
                                             7E  11 001DE      BRB      31$                                        ; 1618
                           57                1E  D0 001E0 28$: MOVL     #30, XPOSCOUNT                             ; 1670
                           5B                27  D0 001E3      MOVL     #39, XPOSBAR                               ; 1671
                           86      461B      8F  B0 001E6      MOVW     #17947, (POINTER)+                         ; 1672
                        41 A8                02  90 001EB      MOVB     #2, 65(R8)                                 ; 1674
              59 00000000G 8F                01  C3 001EF      SUBL3    #1, #FIRST_DATA_LINE, YPOS                 ; 1676
                                             58  11 001F7      BRB      30$                                        ;
                           50        FF      A9  9E 001F9 29$: MOVAB    -1(R9), R0                                 ; 1679
              4C 00000000' EF                50  E1 001FD      BBC      R0, SCR_DATA_LINE, 30$                     ;
                           86      591B      8F  B0 00205      MOVW     #22811, (POINTER)+                         ; 1682
                           86                59  90 0020A      MOVB     YPOS, (POINTER)+                           ; 1683
                           86                57  90 0020D      MOVB     XPOSCOUNT, (POINTER)+                      ; 1684
              66 00000000' EF                09  28 00210      MOVC3    #9, COUNTSTR, (POINTER)                    ; 1686
                           56                09  C0 00218      ADDL2    #9, POINTER                                ; 1687
                           86      591B      8F  B0 0021B      MOVW     #22811, (POINTER)+                         ;
                           86                59  90 00220      MOVB     YPOS, (POINTER)+                           ; 1688
                           86                5B  90 00223      MOVB     XPOSBAR, (POINTER)+                        ; 1689
    86        18           00 00000000'      EF  F0 00226      INSV     REPTSTR, #0, #24, (POINTER)+               ; 1691
                           56                02  C0 0022F      ADDL2    #2, POINTER                                ; 1692
                           86 00000000G      00  90 00232      MOVB     BARCHAR, (POINTER)+                        ;
                           86      4B1B      8F  B0 00239      MOVW     #19227, (POINTER)+                         ; 1693
                 00000000G 8F                59  D1 0023E      CMPL     YPOS, #FIRST_DATA_LINE                     ; 1695
                                             0A  12 00245      BNEQ     30$                                        ;
              50           56                6A  C3 00247      SUBL3    (R10), POINTER, R0                         ; 1696
         40   A8           50        41      A8  83 0024B      SUBB3    65(R8), R0, 64(R8)                         ;
              A0           59 00000000G      8F  F3 00251 30$: AOBLEQ   #LAST_DATA_LINE, YPOS, 29$                 ; 1676
                           86      471B      8F  B0 00259      MOVW     #18203, (POINTER)+                         ; 1701
              68           56                6A  C3 0025E 31$: SUBL3    (R10), POINTER, (R8)                       ; 1709
                           50 00000000G      00  D0 00262 32$: MOVL     NORMAL, R0                                 ; 1711
                                                 04 00269      RET                                                 ; 1712

; Routine Size:  618 bytes,    Routine Base:  $CODE$ + 0000
;  585          1713  1 GLOBAL ROUTINE OUTPUT( STRING ) =
;  586          1714  2 BEGIN
;  587          1715  2
;  588          1716  2 !++
;  589          1717  2 !
;  590          1718  2 ! FUNCTIONAL DESCRIPTION:
;  591          1719  2 !
;  592          1720  2 !       Routine to output counted string with no carriage control.
;  593          1721  2 !
;  594          1722  2 ! INPUTS:
;  595          1723  2 !
;  596          1724  2 !       STRING - address of counted ascii string.
;  597          1725  2 !
;  598          1726  2 ! OUTPUTS:
;  599          1727  2 !
;  600          1728  2 !       none
;  601          1729  2 !++
;  602          1730  2
;  603          1731  2 PUT_TO_SCREEN (.(.STRING)<0,8>, .STRING+1)
;  604          1732  1 END;



                                       0000 00000          .ENTRY  OUTPUT, Save nothing                          ; 1713
               7E       04  AC           01  C1 00002          ADDL3   #1, STRING, -(SP)                             ; 1731
                            7E      04   BC  9A 00007          MOVZBL  @STRING, -(SP)                                ;
           00000000G  00                 02  FB 0000B          CALLS   #2, PUT_TO_SCREEN                             ;
                                             04 00012          RET                                                   ; 1732

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 026A


;  605          1733  1
;  606          1734  1
;  607          1735  1 ROUTINE POSITION( YPOS , XPOS ) =
;  608          1736  2 BEGIN
;  609          1737  2
;  610          1738  2 !++
;  611          1739  2 !
;  612          1740  2 ! FUNCTIONAL DESCRIPTION:
;  613          1741  2 !
;  614          1742  2 !       Routine to call SCRPKG to position screen for characters.
;  615          1743  2 !
;  616          1744  2 ! INPUTS:
;  617          1745  2 !
;  618          1746  2 !       YPOS - y position ( row number , one origin)
;  619          1747  2 !       XPOS - x position ( column number , one origin)
;  620          1748  2 !
;  621          1749  2 ! OUTPUTS:
;  622          1750  2 !
;  623          1751  2 !       none
;  624          1752  2 !--
;  625          1753  2
;  626          1754  2 SCR$SET_CURSOR (.YPOS, .XPOS)   ! set cursor to the requested position
;  627          1755  1 END;
```

```
_$

Sy
--
CA
CA
CH

CH

CH

CH
CH
CH
CH
CH
CH
CH
CJ
CJ
CJ
CJ
CL
CL

CL
CO
CO

CT
CT
CT

CT
CT
CT
CU
CU
DA
DA
DE

DE
DE
DE
DE

DE
DE

DE
```

```
                                            0000 00000 POSITION:
                                                              .WORD   Save nothing                                        ; 1735
                                7E        04    AC  7D 00002  MOVQ    YPOS, -(SP)                                         ; 1754
                       00000000G 00             02  FB 00006  CALLS   #2, SCR$SET_CURSOR                                  ;
                                                    04 0000D  RET                                                         ; 1755

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 027D

;  628          1756  1
;  629          1757  1
;  630          1758  1 END                                  !End of module
;  631          1759  0 ELUDOM



;                              PSECT SUMMARY
;
;       Name                      Bytes                            Attributes
;
;  $OWN$                            147  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                           436  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                           651  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  .  ABS  .                          0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)



;                        Library Statistics
;
;                                            -------- Symbols --------      Pages       Processing
;       File                                 Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1             18619         5         0      1000          00:01.9



;                               COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:TEMPLATE/OBJ=OBJ$:TEMPLATE MSRC$:TEMPLATE/UPDATE=(ENH$:TEMPLATE)

; Size:           651 code + 583 data bytes
; Run Time:          00:33.4
; Elapsed Time:      01:07.0
; Lines/CPU Min:      3157
; Lexemes/CPU-Min: 40116
; Memory Used:  351 pages
; Compilation Complete
```

```
_$
Sy
--

GE
GE

GE
GE
GE
GE
GE
GL
GL
HE

HO
HO


IN
IN
IN
IO

IO
IO
IO

IO
IO
IO
IO
IO
IO
IO
JR
JR
JR
JR
KE
KE
LA
LA

LA
LA
LA
LC
LC
LE
LI
LI
LI
LI
```

0243 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY